Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Отчет по задачам №2

на тему: «Методы поиска»

Выполнил: студент группы БВТ1903

Белов Сергей Павлович

Проверил:

Павликов Артём Евгеньевич

# Парламент

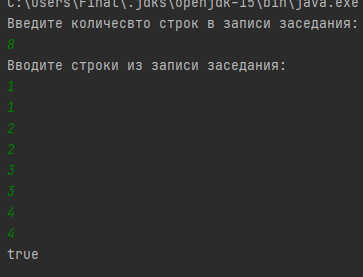
Задача: Вам дана запись о событиях на сессии парламента. Каждое событие является либо внесением нового законопроекта, либо голосованием за какой-то законопроект, причём в обоих случаях известен номер партии, этот законопроект предложившей. События даны в том порядке, в котором они происходили. Проверьте, может ли данная запись о событиях соответствовать какому-либо заседанию, удовлетворяющему порядку проведения, описанному выше.

Алгоритм: если заседание открывается, то к концу оно должно быть закрыто. Используем добавление в стек для добавления голосования, а потом убираем из стека, когда голосование заканчивается. Если в стеке есть что-то, то заседание не может существовать.

Листинг программы:

public class Ex1 {  
 public static void main(String[] arg) throws IOException {  
 //экземпляры классов для считывания  
 Scanner sc = new Scanner(System.*in*);  
 BufferedReader reader = new BufferedReader(new InputStreamReader(System.*in*));  
 //Вводим кол-во строк  
 System.*out*.println("Введите количесвто строк в записи заседания:");  
 int number = sc.nextInt();  
 if(number%2!=0){  
 System.*out*.println("В записи заседания должно быть четное количество записей! Иначе, запись неверна.");  
 }else {  
 //создаем и заполняем массив строк (запись заседания)  
 String[] str = new String[number];  
 System.*out*.println("Вводите строки из записи заседания:");  
 for (int i = 0; i < number; i++) {//вводим строки  
 str[i]=reader.readLine();  
 }  
 System.*out*.println(*Result*(str));  
 }  
 }  
 public static boolean Result (String[] str){  
 //получаем на вход массив строк  
 Stack <String> stack = new Stack<>();  
 //push - добавляет элемент на верх стека  
 //pop - удаляет верхний элемент из стека и возвращает его  
 //peek - возвращает верхний элемент стека, но не удаляет его из стека  
 for (int i=0;i< str.length;i++){  
 if(str[i].charAt(0)=='A'){//если это добавление  
 stack.push(str[i].substring(str[i].length() - 1));   
 }else if(str[i].charAt(0)=='V'){//если это принятие решения  
 char ch = stack.pop().charAt(0);  
 if(str[i].charAt(str[i].length()- 1)!=ch){//то сравниваем последнюю закрытую и последнюю открытую "скобки" (партии)  
 return false;//если "скобки" не одинаковы  
 }  
 }  
 }  
 return true;//если все получилось  
 }  
}

Результат работы программы:



# Правильный N-угольник

Задача: Задан правильный N-угольник. Требуется выбрать наименьшее количество его вершин, которые также образуют правильный многоугольник.

Алгоритм: требуется найти наименьший общий делитель исходного количества углов.

Листинг программы:

public class Ex2 {  
 public static void main (String[] args){  
 int N;//число вершин  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.print("Введите N: ");  
 N = in.nextInt();  
 for (int i=3;i<N+1;i++){//ищем наименьший делитель числа  
 if (N%i==0) {  
 System.*out*.print("наименьшее количество вершин: "+i);  
 break;  
 }  
 }  
 }  
}

Результат работы программы:

